New goals for software tools in language documentation

Kavon Hooshiar - University of Hawai‘i at Mānoa

March 2nd, 2019 - ICLDC 6

Preface

What problem am I looking to solve?

How to build software tools that facilitate and augment the processes that occur between recording language data and analyzing it linguistically.

Context

My perspective/bias/experience: a linguistics student and outsider doing fieldwork with communities speaking under-described languages

This talk builds on the goals for building collection management tools explained in Holton, Hooshiar, and Thieberger (2017)

Why is this still a problem to solve?

Tools for recording language data and analyizing that data are relatively well developed.

Tools for everything in between (metadata creation, file management, archiving, data citation, collaboration, sharing data among language communities) are less developed.

based on criteria of

- user numbers (Hooshiar, Holton, and Thieberger 2017)
- cross-platform availability
- adoption of browser based solutions
- use of open source licenses

Why is this still a problem to solve?

We are good at developing software for functionalities

We are worse at developing software for users

Overview: goals vs functionalities

Goals as important as functionalities

- User experience (what do your users want)
- User interface (what do your users expect)
- Building tools in the framework of existing software standards outside of linguistics
- Open software development and education
- Futureproofing

Functionalities to focus on

- Data standards
- File management
- Version control
- Serving data

High level goals in detail

User experience
(what do your users want)

User experience means: a user's

- emotions
- beliefs
- preferences
- perceptions
- physical and psychological responses

during use of the software (ISO 2010)

This should be the top priority, rather than ancillory, an after thought, or outside the project scope based on funding.

User interface
(what do your users expect)

As dynamic, browser-based applications become ubiquitious, they become the norm for user interaction with applications

Millenials were raised with smart phones and the dynamic web

Traditional software asks the user to "tinker," with many functionalities hidden among menus and settings

Modern software asks the user to focus on content creation, and decreases the user's micromanagement of the application

Modern UI's are not just a plus, they are a necessity

Building on existing infrastructure

- Avoid reinventing the wheel
- User's currently prefer data management software not specific to linguistics (Hooshiar, Holton, and Thieberger 2017)
- We still need language data-specific functionalites
- Today's web is based on layered developments
    - the native power of HTML5 (Rau 2016)
    - fork an open source repo (Forkel 2014)
    - build a plugin (Shepard 2014)

The need for open source

Reciting the need for open source software is now common, but where is the value?

Few of us actually want to interact with the code in a GitHub repository

The need for open source

Open source software protects the software from its creators

Open source software is necessary for ethical concerns

- Big data vs Your data
- Only via open source can we guarantee the moderation of big data approaches to linguistic analysis (cf. Te Reo Māori Example, Te Taka Keegan, this conference)

"Trickle down education" of software users, where few people need ever access the raw code

Futureproofing

How can we mitigate the labor entailed by obselence, and backwards compatibility?

Conforming to data standards

Functionalities in detail

What is data

Archivist perspective:

- Data: recordings of language acts
- Metadata: descriptions of that language act
- Analysis: descriptions of the language contained in that language act

I adopt the following modification

- Data: recordings of language acts
- Metadata: all descriptions of the data (collapse metadata and analysis)

What is data

- Data: Recordings of language acts
- Metadata: All descriptions of the data (collapse metadata and analysis)

This distinction informs software goals

- language acts vs descriptions
- data files vs plain text files
- large files vs small files

Toward data structure standards

A data structure that contains all metadata, and references to data

Can be conceptualized as time aligned tiers

Can be accomplished with nested key:value pairs (e.g. JSON)

Toward data structure standards

Allows for

- iterative adoption of common fields (keys) as a standard
- versatility of developing new keys for different communities (e.g. languages, subfields)

Issues specific to language data

So far we've seen the value of existing infrastructure (the dynamic web, GitHub, JSON data format)

Why not just build a website?

Why not just commit to GitHub?

Issues specific to language data

Issues with an ordinary website

- can't have a centralized server for data management
  - ethical standards of data ownership
  - academic standards of metadata management
  - reliable/affordable access to the internet not ubiquitous

Issues with repositories (“Recommended Data Repositories” 2018)

- lack of (non-language) archives that allow for
  - large files
  - controlled access

Solution through file management software

Build a cross-platform desktop application/downloadable app that manages browser-based access to local files (similar to Dropbox, Drive, etc)

Third-party software development achieved through access to this application

Adherence to the data standards makes import/export of metadata trivial

Git for versioning

Language data projects/collections are contained in repositories

Repository brances allow for

- collaboration (cf. development branch (Driessen 2010))
- archiving (cf. release branch)
- reproducibility (cf. cloning)

Allows for offline versioning

- Note the difference between Git (a distributed version control system) and GitHub (an online host for git repositories)

Use of standard web data formats mitigates diff issues (Bradley McDonnell, personal communication)

Archives as servers

Accessibility brings the archive to life

- Already achieved for researchers interested in reproducibility
- Lacking for language communities interested in sharing data

An API for archives would transform them into the cornerstone of the workflow

- Replaces the centralized server of an ordinary website
- Solves ethics issue of community access

Examples:

- Integrate the API access into Mukurtu
- Host your own site with negligible server costs

Closing thoughts

On the word "new" in my title

Okay, you got me. "New" was a buzz word to lure you to my talk.

Fortunately, many of the items discussed above are in the literature.

Even better, we are seeing many examples of working toward these goals at this conference.

And yet, much remains to be done.

Current research

Currently running a survey about the use of software tools by language community members, linguists, archivists, etc.

The goal is to improve the user experience of new tools based on feedback about current tools

Please take the survey at langit.org, and spread the word!

Takes 5-10 min

Conclusions

To summarize the development model proposed in this talk:

We need effortless metadata (in the classic sense) creation, one-touch archiving, versioning in archives, seamless communication between independent analysis tools, and archives that serve community websites.

Coordinated efforts by independent developement teams that share standards and goals can get us there.

Mahalo(s)

Survey at langit.org

References

Driessen, Vincent. 2010. “A Successful Git Branching Model.” *Nvie.com*. http://nvie.com/posts/a-successful-git-branching-model/.

Forkel, Robert. 2014. “The Cross-Linguistic Linked Data Project.” In *Proceedings of the 3rd Workshop on Linked Data in Linguistics: Multilingual Knowledge Resources and Natural Language Processing*. Reykjavik, Iceland.

Holton, Gary, Kavon Hooshiar, and Nick Thieberger. 2017. “Developing Collection Management Tools to Create More Robust and Reliable Linguistic Data.” In *ComputEL-2*. Honolulu, Hawai‘i.

Hooshiar, Kavon, Gary Holton, and Nicholas Thieberger. 2017. “Survey Results - Linguistic Data Management Survey.” In *MEaCOM Workshop 1*. Alcanena, Portugal.

ISO. 2010. “ISO 9241-210:2010.” *ISO*. http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/05/20/52075.html.

Rau, Felix. 2016. “CMDI Maker - the State and Prospects of a HTML5 Web App.” In. Melbourne.

“Recommended Data Repositories.” 2018. *Scientific Data*. https://www.nature.com/sdata/policies/repositories.

Shepard, Michael. 2014. “Review of Mukurtu Content Management System.” *Language Documentation & Conservation* 8 (September).